

# Graph-Based Approaches to Clustering Network-Constrained Trajectory Data

Mohamed K. El Mahrsi<sup>1</sup> and Fabrice Rossi<sup>2</sup>

<sup>1</sup> Télécom ParisTech, Département INFRES  
46, rue Barrault 75634 Paris CEDEX 13, France  
[khalil.mahrsi@telecom-paristech.fr](mailto:khalil.mahrsi@telecom-paristech.fr)

<sup>2</sup> Équipe SAMM EA 4543, Université Paris I Panthéon-Sorbonne  
90, rue de Tolbiac 75634 Paris CEDEX 13, France  
[fabrice.rossi@univ-paris1.fr](mailto:fabrice.rossi@univ-paris1.fr)

**Abstract.** Even though clustering trajectory data attracted considerable attention in the last few years, most of prior work assumed that moving objects can move freely in an euclidean space and did not consider the eventual presence of an underlying road network and its influence on evaluating the similarity between trajectories. In this paper, we present two approaches to clustering network-constrained trajectory data. The first approach discovers clusters of trajectories that traveled along the same parts of the road network. The second approach is segment-oriented and aims to group together road segments based on trajectories that they have in common. Both approaches use a graph model to depict the interactions between observations w.r.t. their similarity and cluster this similarity graph using a community detection algorithm. We also present experimental results obtained on synthetic data to showcase our propositions.

**Keywords:** similarity, clustering, moving objects, trajectories, road network.

## 1 Introduction

Recent progress in telecommunications and geo-positioning contributed to the democratization of location-aware devices (GPS, smartphones, PDA, etc.) capable of retrieving, storing and sharing their position. Thanks to these devices, it becomes feasible to construct dedicated systems to store the trajectories of various types of moving objects (vehicles, pedestrians, etc.). The availability of such data has shed the light on new challenges and motivated work on management, analysis and data mining of Moving Object Databases (*MOD*) [1].

One of the domains where mining trajectory data can be beneficially applied is road traffic analysis. In fact, collecting and analyzing the GPS logs of vehicles traveling along the road network in order to deduce the state of the network and understand the flow dynamics can be a more efficient and affordable alternative to the use of dedicated sensors (which are expensive to deploy and maintain).

This paper tackles the problem of clustering trajectories of vehicles moving along a road network. More precisely, we study two separate clustering problems: i. the trajectory-based clustering problem, in which we are interested in discovering groups of trajectories that visited the same parts of the road network; and ii. the segment-based clustering problem, which aims to retrieve clusters of road segments that are co-traveled frequently. Studying those clustering problems can be useful in many contexts, such as:

- Management and planning of road network infrastructure: trajectory clustering gives insight into the way the road network is used and grouping together segments that are co-traveled on regular basis can be very helpful for predicting propagation of congestion situations. This knowledge can be used to guide future planning decisions (construction of new roads and lanes, etc.);
- Carpooling: clusters of trajectories represent opportunities for carpooling which, if seized, can help reduce traveling costs and have a positive environmental impact.

The majority of existing literature consider the case of objects moving freely on the euclidean space, neglecting, in the case of vehicular trajectories, the movement constraints imposed by the underlying road network. Moreover, existing approaches are often very sensitive to their parameter values which should be fine-tuned depending on the dataset at hand to produce relevant clustering results. Our contribution addresses these two concerns and can be summarized in the following points:

- We define a similarity measure that compares trajectories based on their proximity on the road network;
- We propose a graph representation to model interactions between trajectories w.r.t. their similarity.
- We use modularity-based community detection to cluster the similarity graphs and discover hierarchies of nested trajectory clusters suitable for exploration at various levels of detail;
- We proceed in analogy and define a clustering approach that regroups road segments based on common trajectories that travel them;
- We present experimental results to showcase our segment clustering approach on a synthetic dataset.

The rest of this paper is organized as follows. We present our data model and our formulation of the studied clustering problems in Section 2. Section 3 describes our trajectory clustering approach. Our road segment clustering approach is presented in Section 4. Experimental results are shown in Section 5. Related work is discussed in Section 6. Finally, Section 7 concludes the paper.

## 2 Data Representation and Problem Statement

A road network is, the most commonly, represented as a directed graph  $G = (V, E)$  [2,3,4]. The set of nodes, or vertices,  $V$  represents intersections and terminal points of roads whereas the set of directed edges  $E$  represents the road

segments interconnecting those nodes. A directed edge  $e = (v_i, v_j)$  indicates that a road segment links the two nodes  $v_i$  and  $v_j$  and that it can be traveled from  $v_i$  in the direction of  $v_j$  but not the other way around (unless another edge states otherwise).

A constrained trajectory  $T$  that travels along this road network can be modeled as a sequence of connected segments:

$$T = \langle id, \{e_1, e_2, \dots, e_l\} \rangle \quad (1)$$

Where  $id$  is the identifier of the trajectory,  $l$  its length (i.e. number of segments) and  $\forall 1 \leq i < l, e_i$  and  $e_{i+1}$  are connected segments belonging to  $E$ . In a real-case scenario, trajectories are collected as GPS logs (sequences of latitude and longitude points) on which a map matching technique (e.g. [5,3]) is applied in order to produce the sequence of traveled segments. The map matching step is out of the scope of this paper.

Given a set of trajectories  $\mathcal{T} = \{T_1, T_2, \dots, T_n\}$  that traveled along a road network  $G = (V, E)$ , we define the following two clustering problems that we will try to solve afterwards:

- The trajectory-based clustering problem consists in partitioning  $\mathcal{T}$  into a set of clusters  $\mathcal{C} = \{C_1, \dots, C_K\}$  of trajectories that exhibited similar behavior. Resemblance between trajectories of the same cluster  $C_i$  should be as high as possible and trajectories across two different clusters  $C_i$  and  $C_j$  should be as different as possible.
- The segment-based clustering problem, on the other hand, aims to partition the set of road segments  $E$  into clusters of segments that recurrently co-appear in trajectories. It is desirable that: i. if a segment  $s$  in a given cluster  $C_i$  appears in a subset of trajectories in  $\mathcal{T}$  then other segments in  $C_i$  are likely to appear in that same subset; and ii. segments belonging to different clusters are unlikely to appear together very often.

### 3 Trajectory-Based Clustering

In this section, we present our solution to the trajectory-based clustering problem. Our clustering framework can be divided into three steps. First, we introduce a measure to assess the similarity between trajectories based on their proximity on the road network (Section 3.1). Then, we use this measure to build a graph depicting the similarity between trajectories (Section 3.2). This similarity graph is partitioned using a community-detection algorithm (Section 3.3) in order to discover meaningful trajectory clusters.

#### 3.1 Similarity Between Trajectories

We adopt a bag-of-segments model where each trajectory is considered as an unordered collection of segments. When comparing two trajectories, the presence of each segment is checked individually without accounting for the segment's

order in the trajectory or the presence of other segments. This simplification is justified by two observations: i. even if the order is unaccounted for explicitly, the underlying network model is a directed graph. Consequently, the direction of travel is implicitly respected since the visited edges are not the same for each direction; and ii. in a context of traffic analysis, congestion situations occur first in singular isolated segments and spread afterwards to adjacent segments, considering individual segments as the basis for comparison is the most natural and intuitive choice.

Intuitively, all segments do not have the same discriminative power. Segments that are frequently traveled by the majority of trajectories are not very relevant to cluster formations. On the contrary, segments that are traveled by a small portion of trajectories play a key role in the formation of the cluster containing those trajectories. To account for this observation, we devise a segment weighting strategy by adapting the TF-IDF (Term Frequency - Inverse Document Frequency) weighting to the case of trajectory data.

We define the spatial segment frequency (ssf) to measure the importance of an edge  $e$  in a trajectory  $T$ :

$$\text{ssf}_{e,T} = \frac{n_{e,T} \cdot \text{length}(e)}{\sum_{e' \in T} n_{e',T} \cdot \text{length}(e')} \quad (2)$$

$n_{e,T}$  being the number of occurrences of  $e$  in  $T$  ( $n_{e,T} = 1$  most of the time as trajectories rarely visit a segment more than once) and  $\text{length}(e)$  its spatial length.

The inverse trajectory frequency (itf) measures the frequency of the segment  $e$  in the whole set of trajectories  $\mathcal{T}$ :

$$\text{itf}_e = \log \frac{|\mathcal{T}|}{|\{T_i : e \in T_i\}|} \quad (3)$$

$|\mathcal{T}|$  is the total number of trajectories in the set  $\mathcal{T}$  and  $|\{T_i : e \in T_i\}|$  the number of trajectories containing the segment  $e$ . While inspecting trajectory  $T$ , the weight attributed to the road segment  $e$  is the combination of both its ssf in the trajectory and its itf:

$$\omega_{e,T} = \text{ssf}_{e,T} \cdot \text{itf}_e \quad (4)$$

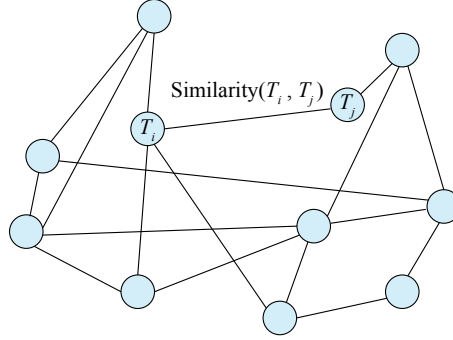
Finally, we compare two trajectories  $T_i$  and  $T_j$  by calculating their cosine similarity:

$$\text{Similarity}(T_i, T_j) = \frac{\sum_{e \in E} \omega_{e,T_i} \cdot \omega_{e,T_j}}{\sqrt{\sum_{e \in E} \omega_{e,T_i}^2} \cdot \sqrt{\sum_{e \in E} \omega_{e,T_j}^2}} \quad (5)$$

### 3.2 Trajectory Similarity Graph

We model the similarity relationships between trajectories as an undirected, weighted graph  $G_S = (\mathcal{T}, E', W)$ . Each trajectory in the set  $\mathcal{T}$  is modeled as a node in  $G_S$ . An edge  $e \in E'$  between a pair of trajectories  $T_i$  and  $T_j$  exists if

and only if  $\text{Similarity}(T_i, T_j) > 0$  (i.e. if they share at least one common road segment). In this case, the similarity is assigned as a weight  $\omega_e \in W$  to the edge  $e$ . The concept of similarity graph is depicted in Fig.1.



**Fig. 1.** Example of a trajectory similarity graph: nodes represent trajectories while edges represent the intensity of the similarity between the trajectories.

In addition to being a natural representation of the interactions between trajectories, the similarity graph puts extra emphasis on the fact that two trajectories that do not share common road segments are totally independent and are not to be "immediately" grouped together in a same cluster. This is reflected in the absence of a similarity edge linking the two trajectories and is fundamentally different from existing approaches that use classic distances and similarities and which can result in regrouping such trajectories together (especially those based on an unconstrained model, which can confuse dissimilar trajectories traveling along two separate but parallel and close roads).

Let  $n = |\mathcal{T}|$  be the number of trajectories and  $m = |E|$  the number of segments in the road network's graph  $G$ . The computational complexity for constructing the similarity graph is  $O(mn^2)$  since it requires  $\frac{n(n-1)}{2}$  similarity calculations each costing  $O(m)$  at the very most.

### 3.3 Clustering the Similarity Graph

Clustering is generally conducted on massive volumes of data. Therefore, the similarity graph described in the previous section tends to have a large number of nodes and edges (especially since only one common road segment is sufficient to create a similarity edge between two trajectories). Modularity-based community-detection algorithms are a popular and widely adopted choice to cluster such graphs [6]. Given a graph  $G = (V, E, W)$ , with vertices  $V = \{v_1, v_2, \dots, v_n\}$ , weighted edges  $E$  such as  $\omega_{ij} \geq 0$  and  $\omega_{ij} = \omega_{ji}$ , and given a partition of the vertices into  $K$  clusters (or communities)  $C_1, \dots, C_K$ , the modularity of the partition is expressed according to the formula:

$$\mathcal{Q} = \frac{1}{2m} \sum_{k=1}^K \sum_{i,j \in C_k} \left( \omega_{ij} - \frac{d_i d_j}{2m} \right) \quad (6)$$

$d_i = \sum_{j \neq i} \omega_{ij}$  and  $m = \frac{1}{2} \sum_i d_i$ . The modularity measures the quality of the clustering by inspecting the arrangement of the edges within the communities of vertices. A high modularity is an indicator that the edges within the communities outnumber (or have higher weights than) those in a similar randomly generated graph (that does not present a community structure). Communities discovered using modularity optimization have a structure that is similar to the structure of cliques. In our case, this means that trajectories grouped together are heavily connected (which is the intended result) and share, two at a time, a considerable number of significant road segments. Moreover, the modularity accounts for the inspected nodes' degrees and can therefore avoid problems that might occur when dealing with lengthy trajectories (that are connected to a great number of other trajectories).

To cluster our similarity graph, we use an implementation of the algorithm described in [7] (the full details of the used implementation as well as the pseudocodes of its different steps can be found in [8]). First, the algorithm finds, for the similarity graph  $G_S$ , the partition  $\mathcal{C} = \{C_1, C_2, \dots, C_K\}$  that has the highest modularity. The significance of this partition is then evaluated by comparing its modularity to the modularity of optimal partitions obtained on similar randomly generated graphs. If the partition is valid (i.e. presents indeed a community structure), then the clustering algorithm is applied recursively on each of its clusters: for each cluster  $C_i \in \mathcal{C}$ , the sub-graph containing only nodes in  $C_i$  and their similarity edges is isolated and clustered separately. The recursion stops once no further significant partitions can be found. The result of this clustering step is a complete hierarchy of nested trajectory clusters that can be explored at various levels of detail.

Since the similarity graph contains  $n$  nodes and, at most,  $\frac{n(n-1)}{2}$  edges ( $n$  being the number of trajectories in  $\mathcal{T}$ ), the theoretical (maximal) complexity of the community detection algorithm used in our clustering phase is  $O(n^3)$  as reported in [6] (however, this complexity is rarely observed in practice where the complexity is somewhere near  $O(n^2)$ ).

## 4 Segment-Based Clustering

In this section, we are interested in discovering groups of segments that are very often traveled together. To solve our segment-based clustering problem, we proceed in analogy to the approach presented in the previous section. We define a similarity measure between segments based on comparison of the trajectories that travel them. This similarity is used to construct a similarity graph that is partitioned in order to discover the clusters of segments.

#### 4.1 Similarity Between Road Segments

We consider that two road segments represented by the two directed edges  $e_i$  and  $e_j$  ( $e_i, e_j \in E$ ) are similar if they frequently co-appear in the same trajectories (i.e. there exists a non-empty subset of trajectories in  $\mathcal{T}$ :  $\{T \in \mathcal{T} : e_i \in T \wedge e_j \in T\}$ ). The larger the number of concomitant appearances of both segments is, the more they are considered similar. We can assimilate this concept of segment similarity to considering each segment as the bag-of-trajectories that traveled it. Comparing two road segments is, therefore, equivalent to comparing the collections of trajectories that visited each one of them.

The claim that we advanced in Section 3.1 can be extended to the case of comparing road segments. A lengthy trajectory that visits a considerable number of road segments is not very informative when judging the similarity between two segments in particular and vice versa. Therefore, we assign weights to trajectories depending on their contribution in characterizing road segments. The weight of a trajectory  $T$ , while inspecting a road segment  $e$ , is:

$$\omega_{T,e} = \frac{n_{e,T}}{\sum_{T' \in \mathcal{T}} n_{e,T'}} \cdot \log \frac{|E|}{|e \in E : e \in T|} \quad (7)$$

The first term in this weight calculates the contribution of  $T$  to the segment  $e$  by calculating the ratio between the number of appearances of  $e$  in  $T$  and the total number of appearances of  $e$  in the whole dataset of trajectories  $\mathcal{T}$ . The second term evaluates the general importance of the trajectory and drops as the trajectory visits more segments and vice versa. Here again, we compare segments using their cosine similarity (by analogy to the formula in eq.(5)).

#### 4.2 Constructing and Clustering the Segment Similarity Graph

We define the segment similarity graph in the same fashion that we defined the trajectory similarity graph in Section 3.2. The segment similarity graph  $G_S(E, E', W)$  is an undirected, weighted graph where each node represents a road segment  $e \in E$ . A similarity edge  $e' \in E'$  links two edges  $e_i$  and  $e_j$  if  $\text{Similarity}(e_i, e_j) > 0$  (i.e. if  $\{T \in \mathcal{T} : e_i \in T \wedge e_j \in T\}$  contains at least one trajectory) in which case their similarity is assigned as a weight to  $e'$ .

We refer to this approach of constructing the similarity graph as the "loose" approach. We can also devise a "strict" approach where a similarity edge exists between the two edges  $e_i$  and  $e_j$  not only if  $\text{Similarity}(e_i, e_j) > 0$  but also if  $e_i$  and  $e_j$  are connected (i.e. the end node of one of the segments is the start node of the other). We apply the exact same community-detection algorithm mentioned in Section 3.3 to the segment similarity graph in order to discover the clusters of similar road segments. The theoretical complexities of the graph construction and the clustering steps can be deduced by analogy from those mentioned in Section 3.

## 5 Experimental Results

In this section, we present preliminary results of the segment-based clustering approach on a synthetic dataset simulated using the Brinkhoff generator [9] (the interested reader is referred to [10] where we briefly reported some results of our trajectory-based approach). The dataset was generated using the Oldenburg map, which contains 6105 nodes and 14070 directed edges. It contains 1000 trajectories that traveled along a total of 7890 unique segments.

Applying the loose approach results in a similarity graph containing 572903 edges. Clustering this graph yields a hierarchy of clusters that spans over seven levels, with only 16 discovered clusters at the top-most level and up to 1222 clusters in the bottom level. Clusters found at the highest levels are, generally, very coarse and are hard to interpret as they regroup a very large number of road segments that can be scattered across different regions of the road network (cf. clusters (a) through (c) in Fig.2). Cluster quality rises as clusters split into more meaningful sub-clusters in the sub-sequent levels of hierarchy (clusters (d) through (f) in Fig.2).

The strictly constructed graph, on the other hand, contains only 8674 similarity edges and is clustered by the community into a hierarchy containing only three levels. The top-most level is composed of 92 levels whereas the third level contains 216 clusters. Fig.3 showcases some of the top level clusters obtained by application of the strict segment clustering.

For comparison's sake, we tried clustering the road segments using classic hierarchical agglomerative clustering (HAC) with average linkage. However, this approach resulted in very disproportionate clusters of poor quality (visualization of resulting clusters is omitted due to space limitation). The reason is that, during the first merging steps, the HAC can make some poor choices that are not reconsidered in the following steps. This limitation is overcome by the community detection algorithm used during our clustering step (cf. Section 3.3) which can permute cluster participants if it can positively contribute to the modularity of the resulting partition.

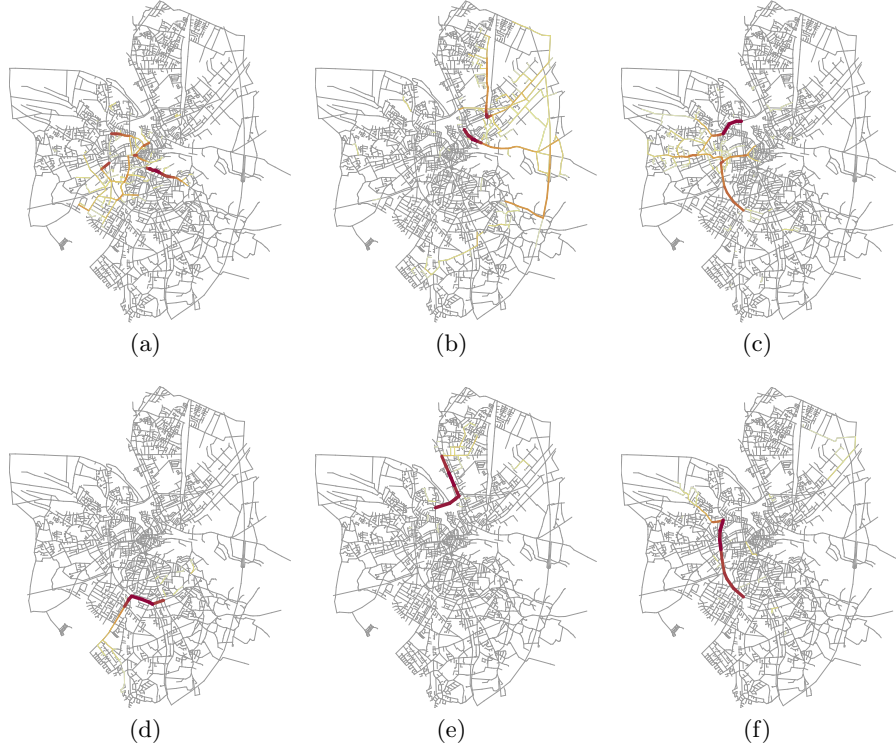
Further experimentations are needed in order to compare clusters produced by the loose approach to those produced by the strict approach and objectively evaluate their quality.

## 6 Related Work

Prior work can be divided into two main areas: i. study of trajectory distances and similarity measures; and ii. design of trajectory clustering algorithms.

Distance measures proposed for free flow trajectories include Dynamic Time Warping (DTW) [11], LCSS (Longest Common Subsequence) [12], ERP (Edit distance with Real Penalty) [13], EDR (Edit Distance on Real sequence) [14] and the One-Way Distance (OWD) [15]. The above distances consider trajectories as sequences of points on the euclidean space and ignore any network-related constraints that can restrict the movement of the studied objects. Therefore, they

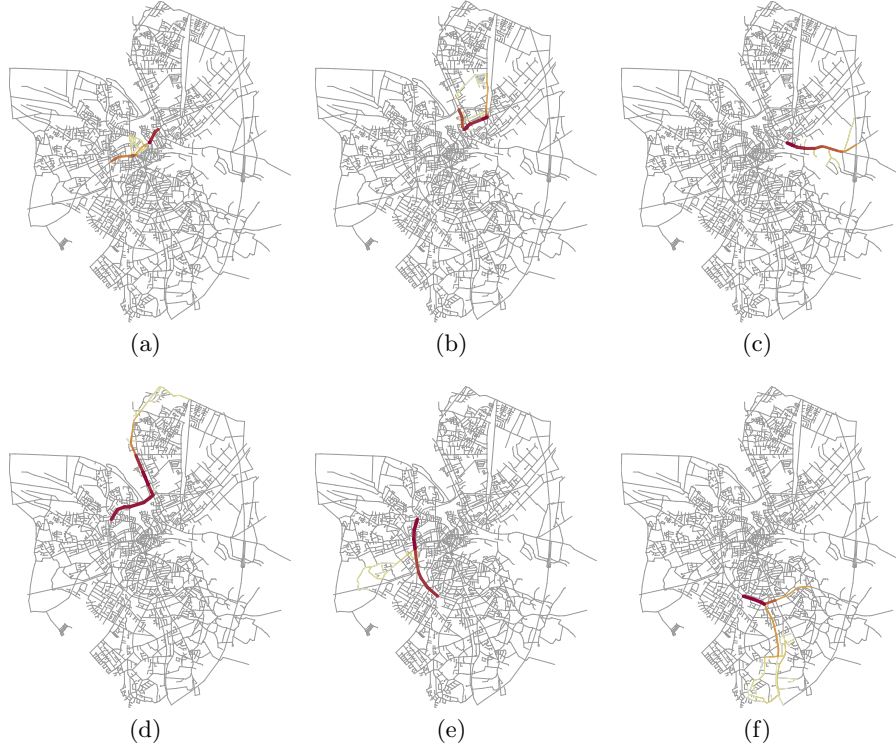




**Fig. 2.** Some clusters of segments discovered through clustering of the loose similarity graph. Clusters depicted in (a), (b) and (c) are top-level, very coarse clusters. Clusters (d) through (f) illustrate how clusters get refined when navigating down the hierarchy of clusters (in this case, the clusters are issued from the second level). The segments' coloration is relative to the number of trajectories traveling them, varying from pale yellow (for less traveled segments) to dark red (for segments that are traveled frequently).

cannot be applied directly in our context. A constrained approach is presented in [16]. However, it requires a priori definition of points of interest in the road network and can't be used for unsupervised learning. Other propositions can be found in [17] and [18].

Approaches to trajectory clustering are adaptations of existing algorithms. Existing problem formulations and propositions include flock patterns [19], convoy patterns [20], the TRACCLUS partition-and-group framework [21] and the T-OPTICS and TF-OPTICS algorithms [22]. The aforementioned algorithms use euclidean-based similarities and distances and can, therefore, be used only in the case of unconstrained trajectories. Furthermore, the majority of these approaches use density-based algorithms which suffer from two major drawbacks: i. their results are very sensitive to the parameter values; and ii. they assume



**Fig. 3.** Some of the clusters discovered through clustering of the strict similarity graph.

that trajectories in the same cluster have a rather homogeneous density, which is rarely the case (as discussed in [4]).

In [2], the authors describe an approach to discovering "dense paths" or sequences of frequently traveled segments in a road network. This approach resembles our segment-based clustering although they diverge on many key aspects. For instance, the approach in [2] produces flat clusters using a density-based approach (which requires fine tuning) whereas ours produces a hierarchy of nested clusters and does not require parametrization.

The most relevant work to our trajectory-based clustering is the one presented by Roh et Hwang in [4] where the distance between trajectories in the road network is measured using shortest path calculations. A baseline algorithm, using agglomerative hierarchical clustering, as well as a more efficient algorithm, called NNCluster, are presented for the purpose of regrouping the network constrained trajectories. Unlike our approach, the distance in [4] can be computationally prohibitive (especially in large road networks) besides from being direction-independent (i.e. it does not take into account the traveling direction, thus requiring an additional filtering step to separate inverted trajectories regrouped together).

## 7 Conclusion

In this paper, we presented two approaches to cluster trajectory data under constraints of an underlying road network. The first approach focuses on clustering entire trajectories into communities of trajectories that exhibited similar behavior and movement patterns. The second approach deals with road segments and tries to discover groups of segments that are often visited concomitantly. Both approaches are based on a two-steps framework. First, they start by computing a similarity graph between the individuals to be clustered (i.e. trajectories or road segments). Then, the graph is clustered using a hierarchical community-detection algorithm.

This framework presents many advantages: i. it does not require parameters, contrary to the majority of existing approaches that are very sensitive to their threshold values; and ii. it also produces a hierarchy of nested clusters promoting exploration at various levels of granularity and detail in situations where a flat clustering approach would have produced a unique level containing a very large number of clusters. However, the framework is not flawless: the community detection algorithm used in the clustering step can be sensitive in presence of noise (i.e. trajectories that do not necessarily belong to any cluster or segments that are traveled rarely) which can degrade the quality of the discovered clusters.

For future work, we would like to conduct more thorough experiments with our segment-based approach and evaluate the quality of the produced clusters using internal and external quality indexes. We would also like to test our approaches on real vehicle trajectory datasets and study the impact of varying the community detection algorithm used in the clustering phase.

## References

1. Giannotti, F., Pedreschi, D., eds.: *Mobility, Data Mining and Privacy - Geographic Knowledge Discovery*. Springer (2008)
2. Kharrat, A., Popa, I.S., Zeitouni, K., Faiz, S.: Clustering algorithm for network constraint trajectories. In Ruas, A., Gold, C.M., eds.: *SDH. Lecture Notes in Geoinformation and Cartography*, Springer (2008) 631–647
3. Lou, Y., Zhang, C., Zheng, Y., Xie, X., Wang, W., Huang, Y.: Map-matching for low-sampling-rate gps trajectories. In: *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems. GIS '09*, New York, NY, USA, ACM (2009) 352–361
4. Roh, G.P., Hwang, S.w.: Nncluster: An efficient clustering algorithm for road network trajectories. In Kitagawa, H., Ishikawa, Y., Li, Q., Watanabe, C., eds.: *Database Systems for Advanced Applications. Volume 5982 of Lecture Notes in Computer Science*. Springer Berlin - Heidelberg (2010) 47–61
5. Brakatsoulas, S., Pfoser, D., Salas, R., Wenk, C.: On map-matching vehicle tracking data. In: *Proceedings of the 31st international conference on Very large data bases. VLDB '05, VLDB Endowment* (2005) 853–864
6. Fortunato, S.: Community detection in graphs. *Physics Reports* **486**(3-5) (2010) 75–174

7. Noack, A., Rotta, R.: Multi-level algorithms for modularity clustering. In: Proceedings of the 8th International Symposium on Experimental Algorithms. SEA '09, Berlin, Heidelberg, Springer-Verlag (2009) 257–268
8. Rossi, F., Villa-Vialaneix, N.: Représentation hiérarchique d'un grand réseau à partir d'une classification hiérarchique de ses sommets. *Journal de la Société Française de Statistique* **152** (In press 2011) 34–65
9. Brinkhoff, T.: A framework for generating network-based moving objects. *Geoinformatica* **6** (June 2002) 153–180
10. El Mahrsi, M.K., Rossi, F.: Modularity-Based Clustering for Network-Constrained Trajectories. In: Proceedings of the 20-th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN 2012), Bruges, Belgique (April 2012) 471–476
11. Berndt, D.J., Clifford, J.: Finding patterns in time series: a dynamic programming approach. (1996) 229–248
12. Vlachos, M., Gunopoulos, D., Kollios, G.: Discovering similar multidimensional trajectories. In: ICDE '02: Proceedings of the 18th International Conference on Data Engineering, Washington, DC, USA, IEEE Computer Society (2002) 673
13. Chen, L., Ng, R.: On the marriage of lp-norms and edit distance. In: VLDB '04: Proceedings of the Thirtieth international conference on Very large data bases, VLDB Endowment (2004) 792–803
14. Chen, L., Özsu, M.T., Oria, V.: Robust and fast similarity search for moving object trajectories. In: SIGMOD '05: Proceedings of the 2005 ACM SIGMOD international conference on Management of data, New York, NY, USA, ACM (2005) 491–502
15. Lin, B., Su, J.: Shapes based trajectory queries for moving objects. In: GIS '05: Proceedings of the 13th annual ACM international workshop on Geographic information systems, New York, NY, USA, ACM (2005) 21–30
16. Hwang, J.R., Kang, H.Y., Li, K.J.: Spatio-temporal similarity analysis between trajectories on road networks. In Akoka, J., Liddle, S.W., Song, I.Y., Bertolotto, M., Comyn-Wattiau, I., Cherfi, S.S.S., van den Heuvel, W.J., Thalheim, B., Kolp, M., Bresciani, P., Trujillo, J., Kop, C., Mayr, H.C., eds.: ER (Workshops). Volume 3770 of Lecture Notes in Computer Science., Springer (2005) 280–289
17. Tiakas, E., Papadopoulos, A.N., Nanopoulos, A., Manolopoulos, Y., Stojanovic, D., Djordjevic-Kajan, S.: Trajectory similarity search in spatial networks. In: Proceedings of the 10th International Database Engineering and Applications Symposium, Washington, DC, USA, IEEE Computer Society (2006) 185–192
18. Chang, J.W., Bista, R., Kim, Y.C., Kim, Y.K.: Spatio-temporal similarity measure algorithm for moving objects on spatial networks. In: Proceedings of the 2007 international conference on Computational science and its applications - Volume Part III. ICCSA'07, Berlin, Heidelberg, Springer-Verlag (2007) 1165–1178
19. Benkert, M., Gudmundsson, J., Hübner, F., Wollé, T.: Reporting flock patterns. In: ESA'06: Proceedings of the 14th conference on Annual European Symposium, London, UK, Springer-Verlag (2006) 660–671
20. Jeung, H., Shen, H.T., Zhou, X.: Convoy queries in spatio-temporal databases. In: ICDE '08: Proceedings of the 2008 IEEE 24th International Conference on Data Engineering, Washington, DC, USA, IEEE Computer Society (2008) 1457–1459
21. Lee, J.G., Han, J., Whang, K.Y.: Trajectory clustering: a partition-and-group framework. In: SIGMOD '07: Proceedings of the 2007 ACM SIGMOD international conference on Management of data, New York, NY, USA, ACM (2007) 593–604
22. Nanni, M., Pedreschi, D.: Time-focused clustering of trajectories of moving objects. *J. Intell. Inf. Syst.* **27**(3) (2006) 267–289